Java III

Bem-vindo ao estudo sobre Estruturas de Controle no Java. Este estudo ajudará na compreensão de alguns conceitos e práticas que são importantes no contexto de programação. Vamos conhecer melhor esses conceitos?

Conceitos Iniciais

Neste módulo serão apresentadas a Sintaxe e a Semântica no contexto da programação, como também os comandos e operações das estruturas de decisão e repetição da linguagem de programação Java. Detalhes sobre esses comandos já foram vistos no decorrer desta disciplina, mas na forma de pseudocódigo (algoritmo).

Sintaxe e Semântica

Para projetar um programa seu ciclo de vida começa através de modelos, especificações e por fim o código. Estes modelos e especificações servem para entender e documentar o que um usuário pretende resolver com o programa. Além disso, esta prática facilita muito transformar as ideias em passos, e posteriormente em um algoritmo. Por fim, ser codificado em um programa na forma de uma determinada linguagem de programação.

Sendo assim, os termos sintaxe e semântica fazem parte deste context
onde os códigos dos programas precisam de uma boa forma (sintaxe)

um bom conteúdo (semântica).

A sintaxe geralmente refere-se à forma de escrever código fonte (palavras reservadas, comandos, recursos diversos). Pode-se dizer que, é o conjunto de regras que devem ser seguidas para a escrita de um algoritmo ou programa e tem uma relação direta com a forma (semântica) de como essas regras são descritas (RIBEIRO, 2019).

A semântica é o estudo do significado das coisas (do conteúdo das “formas”). No contexto de programação, refere-se ao significado dos modelos, ao nível de entendimento como: clareza, objetividade, detalhamento, coesão, entre outros (FERREIRA, 1999).

As particularidades da linguagem de programação Java, segundo PUGA e RISSETTI (2016):

- Case Sensitive: Letras maiúsculas se diferenciam das minúsculas Ex.: nome é diferente de NOME ou Nome

- Como em algoritmos há também as palavras reservadas. Que são comandos ou ações e escritas em inglês.

- Comentários podem ser feitos através dos símbolos: /* o que estiver aqui não é executado */ ou // o que estiver na mesma linha não é executado. Servem apenas para informar e organizar o código do programa, o código-fonte.

- Como uma boa prática de programação, abre chaves {temos comandos } fecha chaves para bloco de comandos. Linhas de comandos são fechadas com “;”.

Comandos e Operadores

Os Comandos e Operadores foram apresentados em algoritmos de forma detalhada, nesta seção será mostrado sua equivalência na linguagem de programação Java. Os comandos são as instruções que remetem às ações a serem executadas pelo programa, tais como: comandos de entrada e saída de dados, estruturas de repetição, comandos de decisão, entre outros. Os operadores são utilizados para executar cálculos numéricos e relacionar expressões, são eles: Operadores Aritméticos, Relacionais e Lógicos (PUGA e RISSETTI, 2016). Abaixo, exemplos e equivalências de Operadores e Comandos.

Operadores Aritméticos

- + Adição ou concatenação. Exemplo: 5 + 2 (=7), “Algo” + “ritmo”
- += Adição Exemplo: numero +=2 (numero = numero + 2)
- - Subtração. Exemplo: 5 -3 (= 2)
- -= Subtração. Exempelo: numero -=2 (numero = numero - 2)
- * Multiplicação. Exemplo: 2 * 5 (=10)
- *= Multiplicação. Exemplo: numero *=2 (numero = numero * 2)
- / Divisão. Exemplo de inteiros: 5 / 2 (= 2). Exemplo de reais: 5.0 / 2.0 (= 2.5)
- /= Divisão. Exemplo: numero /=2 (numero = numero / 2)

- % Resto da divisão. Exemplo: 5 % 2 (= 1)
- / Quociente da divisão. Exemplo: 5 / 2 (= 2)

Operadores Relacionais

- = Igual. Exemplo: idade == 20
- != Diferente. Exemplo: idade != 20
- < Menor que. Exemplo: idade < 20
- > Maior que. Exemplo: idade > 20
- <= Menor ou igual que. Exemplo: idade <= 20
- > Maior ou igual que. Exemplo: idade >= 20

Operadores Lógicos

- && E (AND) Exemplo: (idade > 20) && (idade < 50)
- || OU (OR) Exemplo: (idade > 20) || (idade < 50)
- ! Negação Exemplo: !(idade==20)

Estrutura de Decisão

Temos três tipos de estruturas de decisão, a estrutura de decisão simples, a estrutura de decisão composta e a estrutura de decisão encadeada.

Uma estrutura de decisão é utilizada quando apenas uma parte do programa deve ser executado de acordo com uma condição. A parte a ser executada é a que satisfaz determinada condição.

Na estrutura de decisão simples, se a condição for verdadeira, os comandos são executados, caso contrário, nada se faz. Temos a seguinte estrutura:

```
if (<condição)
{
  <comandos>;
}
```

Na estrutura de decisão composta, se a condição for verdadeira, os comandos são executados, caso contrário, outros comandos são executados. Temos a seguinte estrutura:

```
if (<condição)
{
  <comandos>;
}
else
{
  <outros comandos>;
}
```

Na estrutura de decisão encadeada, uma estrutura de decisão simples ou composta faz parte dos comandos a serem executados. Temos a seguinte estrutura:

```
if (<condição)
{
  if (<outra condição>)
    {
      <comandos>;
    }
```

```
}

else

{

  <outros comandos que pode ser outra estrutura de decisão>;

}
```

Outra estrutura de decisão, que denominamos de estrutura de múltipla escolha, você decide por uma das opções e os comandos daquela opção são executadas. Neste caso, a estrutura é apresentada da seguinte forma:

```
switch (<variável>)

{

  case <valor_1> : <comandos1>;

                   break;

  case <valor_2> : <comandos2>;

                   break;

       ...

  case <valor_n> : <comandosn>;

                  break;
```

```
  default : <comandos>;

}
```

Vamos ver um exemplo por meio do desenvolvimento de um programa Java que declara variáveis, recebe uma opção e um número inteiro, calcula se o número é par ou ímpar, positivo ou não positivo e apresenta apenas a opção selecionada. Por fim, apresentar as informações.

Figura 1: Exemplo Olá Mundo em Java

Comandos e Operadores

Os Comandos e Operadores foram apresentados em algoritmos de forma detalhada, nesta seção será mostrado somente sua equivalência na linguagem de programação Java. Os comandos são as instruções que remetem as ações a serem executadas pelo programa, tais como: comandos de entrada e saída de dados, laços de repetição, comandos de decisão, entre outros. Os operadores são utilizados para executar cálculos numéricos e relacionar expressões, são eles: Operadores Aritméticos,

Relacionais e Lógicos (PUGA e RISSETTI, 2016). Abaixo, exemplos e equivalências de Operadores e Comandos.

Operadores Aritméticos

- + Adição ou concatenação. Exemplo: 5 + 2 (=7), “Algo” + “ritmo”
- += Adição Exemplo: numero +=2 (numero = numero + 2)
- - Subtração. Exemplo: 5 -3 (= 2)
- -= Subtração. Exemplo: numero -=2 (numero = numero - 2)
- * Multiplicação. Exemplo: 2 * 5 (=10)
- *= Multiplicação. Exemplo: numero *=2 (numero = numero * 2)
- / Divisão. Exemplo: 5 / 3 (=15)
- /= Divisão. Exemplo: numero /=2 (numero = numero / 2)

Operadores Relacionais

- = Igual. Exemplo: idade == 20
- != Deferente. Exemplo: idade != 20
- < Menor que. Exemplo: idade < 20
- > Maior que. Exemplo: idade > 20
- <= Menor ou igual que. Exemplo: idade <= 20
- > Maior ou igual que. Exemplo: idade >= 20

Operadores Lógicos

- && E (AND) Exemplo: (idade == 20) && (profissao == “professor”)
- || OU (OR) Exemplo: (idade > 20) || (idade < 50)
- ! Negação Exemplo: !(idade==20)

Comando de Entrada de Dados

Através da Biblioteca Scanner é possível receber os valores digitados pelo usuário e incluí-los nas variáveis nome e idade, conforme apresentado na Figura 2. Este comando é equivalente ao comando “LEIA” do algoritmo em pseudocódigo.

```
1  //salvar como ProgDecisao.java
2  import javax.swing.*;
3
4  class ProgDecisao
5  {
6    public static void main (String entrada[])
7    {
8      int num;
9      char op = '0';
10     String msg = "", msgEntr = "Digite 1 para par/impar\nDigite 2 para positivo/nao
       positivo\n";
11     // entrada de dados
12     num = Integer.parseInt(JOptionPane.showInputDialog("Digite um numero inteiro"));
13     op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);
14     // processamento
15
16     switch (op)
17     {
42     //saida de resultados
43     if (op == '1' || op == '2')
44     {
45        JOptionPane.showMessageDialog(null, msg);
46     }
47     System.exit(0);
48   }
49 }
```

Temos uma estrutura de decisão simples da linha 43 à linha 46 do programa. Se o valor da variável op for ‘1’ ou se for ‘2’, então o conteúdo de msg é apresentado, senão nada acontece.

```
16      switch (op)
17      {
18          case '1':
19          {
20              if (num % 2 == 0)
21              {
22                  msg = msg + num + " eh par.\n\n";
23              }
24              else
25              {
26                  msg = msg + num + " eh impar.\n\n";
27              }
28              break;
29          }
30          case '2':
31              if (num > 0)
32              {
33                  msg = msg + num + " eh positivo.\n\n";
34              }
35              else
36              {
37                  msg = msg + num + " eh nao positivo.\n\n";
38              }
39              break;
40          default: JOptionPane.showMessageDialog(null, "Opcao invalida, calculos nao
            realizados");
41      }
```

Temos uma estrutura de decisão composta da linha 20 à 29 e da linha 31 à linha 38 do programa. No primeiro case, se o valor de num for par, concatena a mensagem como sendo par, caso contrário, concatena a mensagem como ímpar. No segundo case, se o valor de num for positivo, concatena a mensagem como sendo positivo, caso contrário, concatena a mensagem como sendo não positivo.

Temos também a estrutura de múltipla escolha do switch/case da linha 16 à 40 que avalia o conteúdo do valor op. Caso for ‘1’ realiza os comandos dentro deste case. Caso for ‘2’ realiza os comandos dentro deste case. Caso nenhum dos case for executado, então o default é executado apresentando a mensagem de opção inválida.

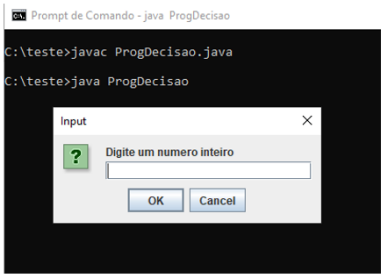
//salvar como ProgDecisao.java

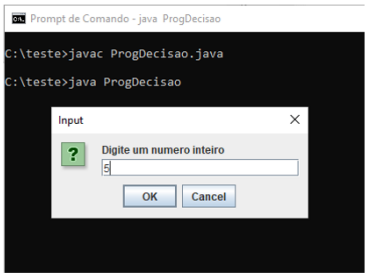
import javax.swing.*;

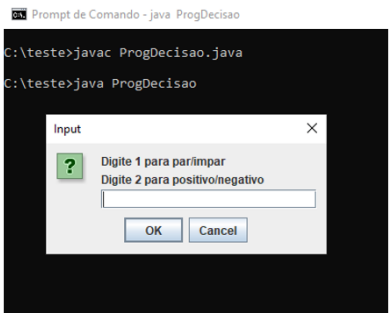
```
class ProgDecisao

{

  public static void main (String entrada[])

  {

        int num;

        char op = '0';

          String msg = "", msgEntr = "Digite 1 para par/impar\nDigite 2 para positivo/nao positivo\n";

        // entrada de dados

            num = Integer.parseInt(JOptionPane.showInputDialog("Digite um numero inteiro"));

        op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);

        // processamento



        switch (op)

        {

              case '1':

              {

                    if (num % 2 == 0)
```

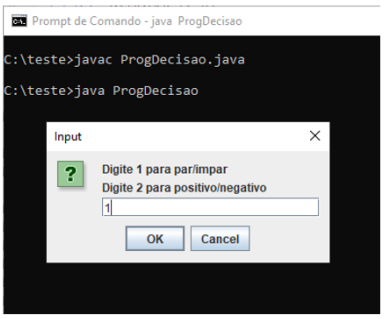
```
            {
                msg = msg + num + " eh par.\n\n";
            }
            else
            {
                msg = msg + num + " eh impar.\n\n";
            }
            break;
        }
        case '2':
            if (num > 0)
            {
                msg = msg + num + " eh positivo.\n\n";
            }
            else
            {
                msg = msg + num + " eh nao positivo.\n\n";
            }
```

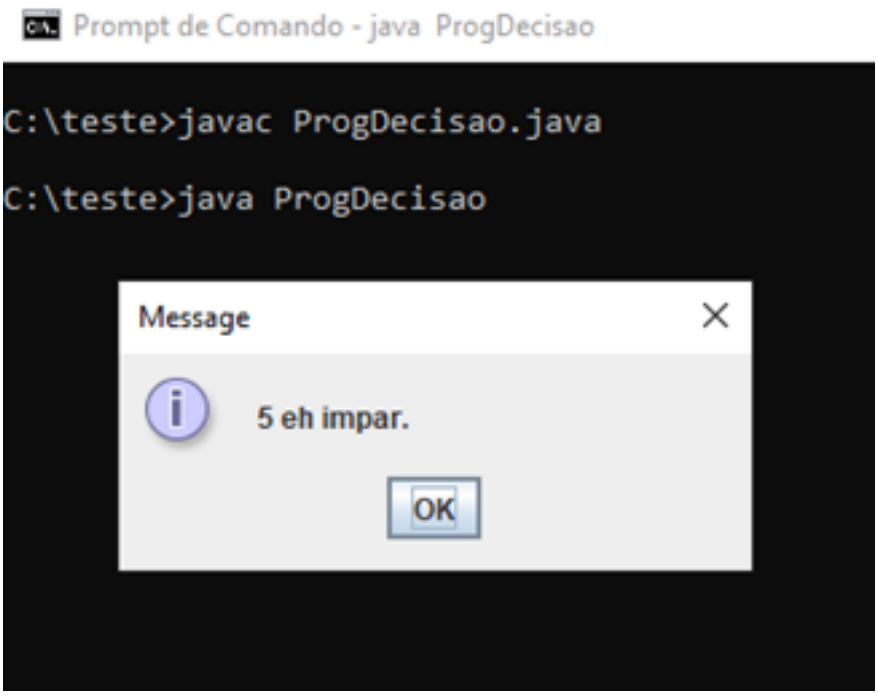
```
                    break;

                    default: JOptionPane.showMessageDialog(null, "Opcao
invalida, calculos nao realizados");

        }

        //saída de resultados

        if (op == '1' || op == '2')

        {

          JOptionPane.showMessageDialog(null, msg);

        }

        System.exit(0);

  }

}
```

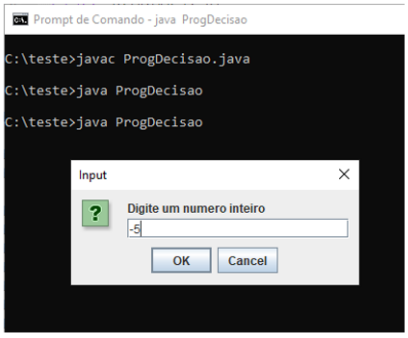

Prompt de Comando - java ProgDecisao
C:\teste>javac ProgDecisao.java
C:\teste>java ProgDecisao
Input
Digite um numero inteiro
OK
Cancel

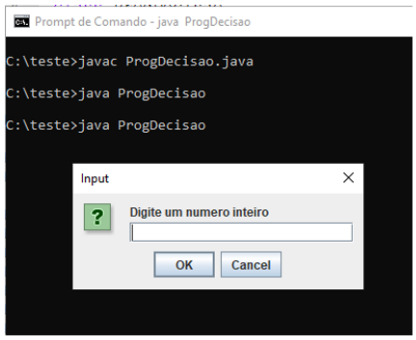

Prompt de Comando - java ProgDecisao
C:\teste>javac ProgDecisao.java
C:\teste>java ProgDecisao
Input
Digite um numero inteiro
5
OK
Cancel

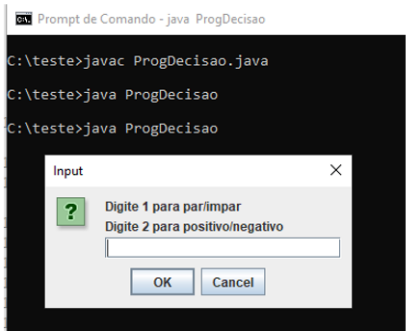

Prompt de Comando - java ProgDecisao
C:\teste>javac ProgDecisao.java
C:\teste>java ProgDecisao
Input
Digite 1 para par/impar
Digite 2 para positivo/negativo
OK
Cancel

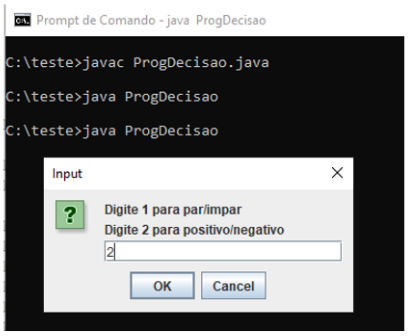

Prompt de Comando - java ProgDecisao
C:\teste>javac ProgDecisao.java
C:\teste>java ProgDecisao
Input
Digite 1 para par/impar
Digite 2 para positivo/negativo
1
OK
Cancel

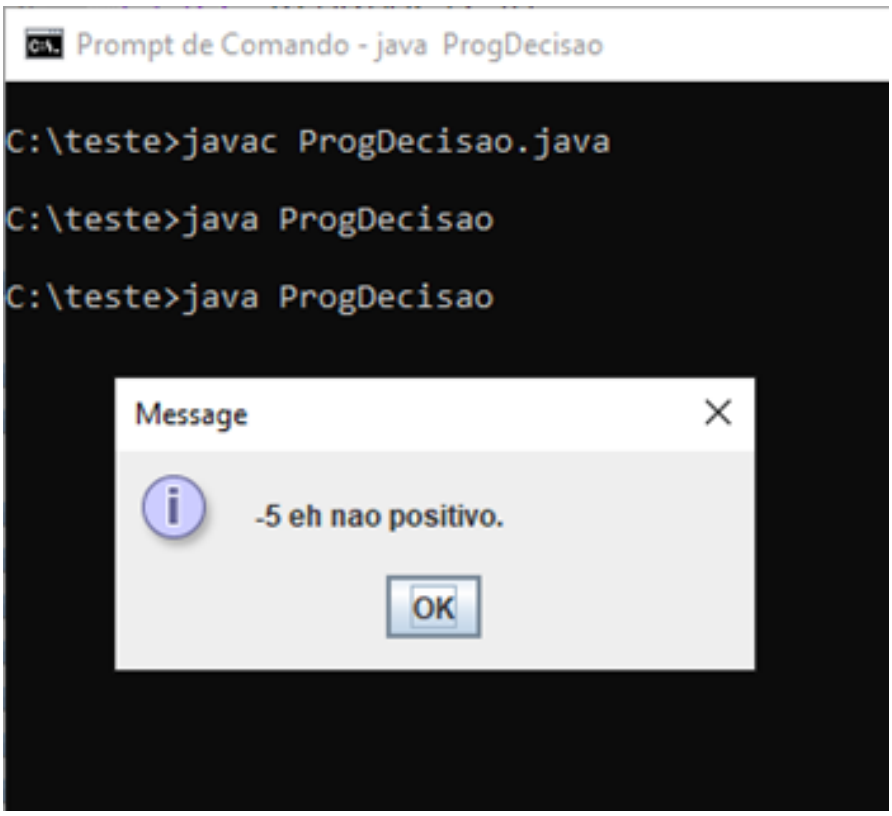

Prompt de Comando - java ProgDecisao
C:\teste>javac ProgDecisao.java
C:\teste>java ProgDecisao
Message
5 eh impar.
OK











Estrutura de Repetição

Temos três tipos de estrutura de repetição, a estrutura de repetição do for, a estrutura de repetição do while e a estrutura de repetição do do/while.

Utilizamos uma estrutura de repetição quando precisamos repetir por diversas vezes um mesmo conjunto de comandos.

Numa estrutura de repetição é importante você garantir quando se inicia a repetição, a condição de parada e o comando de continuação na repetição.

Para a estrutura de repetição do for no java, temos a seguinte estrutura:

for (<comando inicial> ; <condição de parada> ; <comando de continuação>)

{

  <comandos>;

}

Para a estrutura de repetição do while no java, temos a seguinte estrutura:

<condição inicial>;

while (<condição de parada>)

{

```
  <comandos>;

  <condição de continuação>;

}
```

Para a estrutura de repetição do do/while no java, temos a seguinte estrutura:

```
<condição inicial>;

do

{

  <comandos>;

  <condição de continuação>;

} while (<condição de parada>);
```

Para exemplificar, vamos fazer um programa java que declara variáveis, receba um número para calcular a tabuada por alguma dessas estruturas de repetição, mostrando o resultado da tabuada.

```java
//salvar como ProgRepeticao.java
import javax.swing.*;
class ProgRepeticao
{
  public static void main (String entrada[])
  {
    int Tabuada, i;
    char op = '0';
    String msg = "", msgEntr = "Digite 1 para repeticao for\nDigite 2 para repeticao
    while\nDigite 3 para repeticao do/while\n\n";
    // entrada de dados
    Tabuada = Integer.parseInt(JOptionPane.showInputDialog("Digite um numero inteiro"));
    op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);
    // processamento

    switch (op)
    {
    //saida de resultados
    if (op >= '1' && op <= '3')
    {
       JOptionPane.showMessageDialog(null, msg);
    }
    System.exit(0);
  }
}
```

```java
    switch (op)
    {
        case '1':
        {
            msg = msg + "Tabuada do " + Tabuada + " pelo for: \n\n";
            for(i = 1 ; i<=10 ; i=i+1)
            {
                msg = msg + Tabuada + " x " + i + " = " + Tabuada*i + "\n";
            }
            break;
        }
        case '2':
            msg = msg + "Tabuada do " + Tabuada + " pelo while: \n\n";
            i = 1;
            while(i<=10)
            {
                msg = msg + Tabuada + " x " + i + " = " + Tabuada*i + "\n";
                i=i+1;
            }
            break;
        case '3':
            msg = msg + "Tabuada do " + Tabuada + " pelo do/while: \n\n";
            i = 1;
            do
            {
                msg = msg + Tabuada + " x " + i + " = " + Tabuada*i + "\n";
                i=i+1;
            } while(i<=10);
            break;
        default: JOptionPane.showMessageDialog(null, "Opcao invalida, calculos nao
        realizados");
    }
```

Da linha 21 à 24, a estrutura de repetição do for está sendo utilizado para o cálculo da Tabuada, iniciando em 1, terminando em 10 e incrementando de um a um a variável i.

Da linha 29 à 34, a estrutura de repetição do while está sendo utilizado para o cálculo da Tabuada, iniciando em 1, terminando em 10 e incrementando de um a um a variável i.

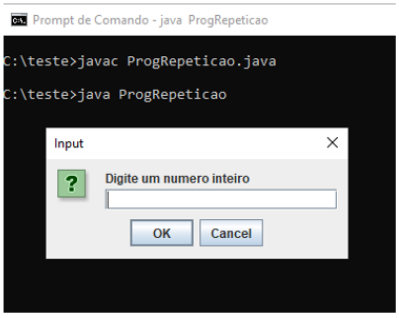
Da linha 38 à 43, a estrutura de repetição do do/shile está sendo utilizado para o cálculo da Tabuada, iniciando em 1, terminando em 10 e incrementando de um a um a variável i.

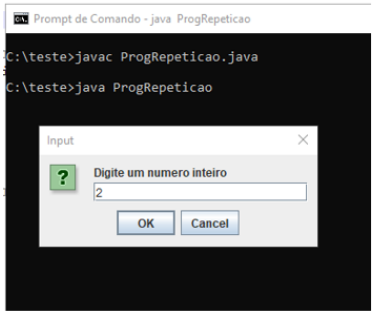
```
//salvar como ProgRepeticao.java

import javax.swing.*;


class ProgRepeticao

{

  public static void main (String entrada[])

  {

        int Tabuada, i;

        char op = '0';

        String msg = "", msgEntr = "Digite 1 para repeticao for\nDigite 2 para repeticao while\nDigite 3 para repeticao do/while\n\n";

        // entrada de dados

            Tabuada = Integer.parseInt(JOptionPane.showInputDialog("Digite um numero inteiro"));

        op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);

        // processamento
```

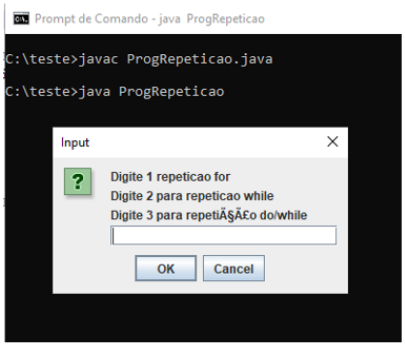
```
        switch (op)
        {
            case '1':
            {
                msg = msg + "Tabuada do " + Tabuada + " pelo for: \n\n";
                for(i = 1 ; i<=10 ; i=i+1)
                {
                    msg = msg + Tabuada + " x " + i + " = " + Tabuada*i
+ "\n";
                }
                break;
            }
            case '2':
                msg = msg + "Tabuada do " + Tabuada + " pelo while:
\n\n";
                i = 1;
                while(i<=10)
                {
```

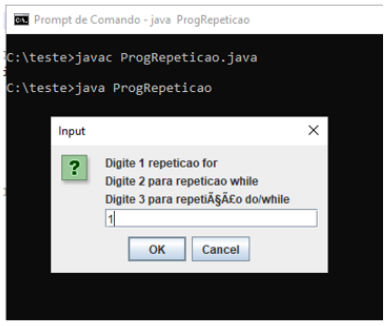
```
                    msg = msg + Tabuada + " x " + i + " = " + Tabuada*i
+ "\n";

                    i=i+1;

                }

                break;

            case '3':

                msg = msg + "Tabuada do " + Tabuada + " pelo do/while:
\n\n";

                i = 1;

                do

                {

                    msg = msg + Tabuada + " x " + i + " = " + Tabuada*i
+ "\n";

                    i=i+1;

                } while(i<=10);

                break;

                default: JOptionPane.showMessageDialog(null, "Opcao
invalida, calculos nao realizados");

        }

        //saída de resultados
```

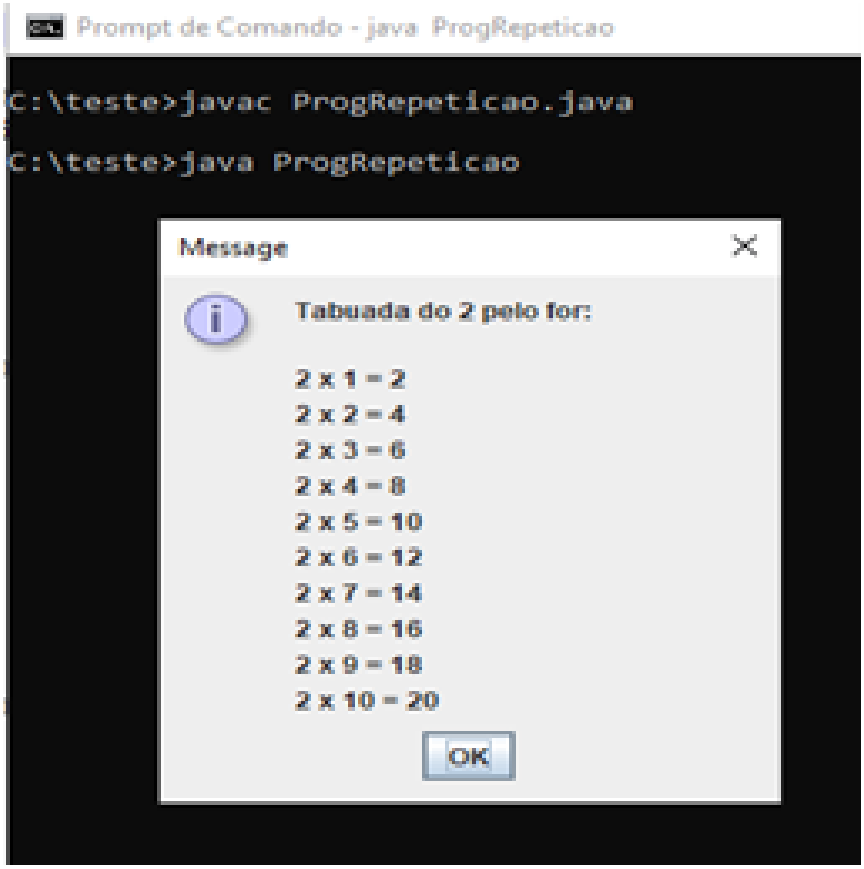
```
        if (op >= '1' && op <= '3')

        {

          JOptionPane.showMessageDialog(null, msg);

        }

        System.exit(0);

    }

}
```

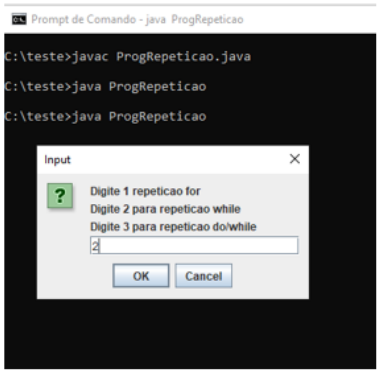


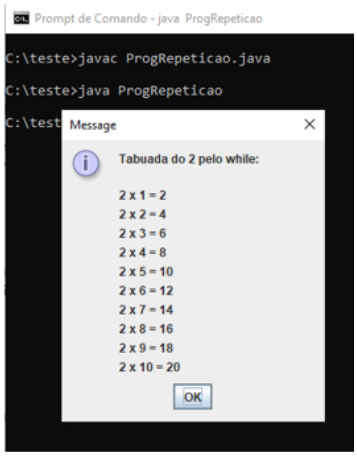


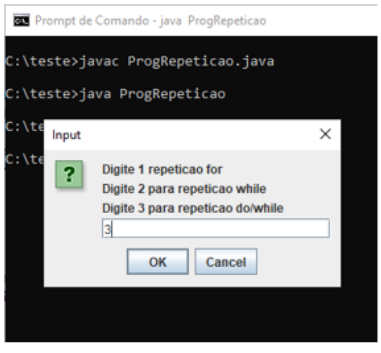


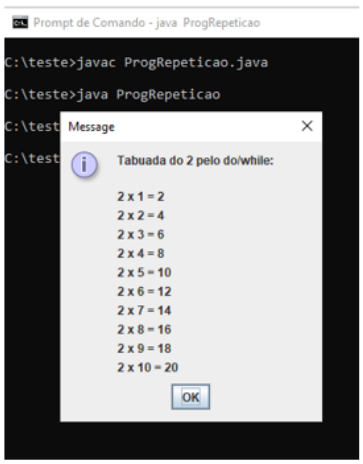



Prompt de Comando - java ProgRepeticao
C:\teste>javac ProgRepeticao.java
C:\teste>java ProgRepeticao
Message
Tabuada do 2 pelo for:
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
OK


Prompt de Comando - java ProgRepeticao
C:\teste>javac ProgRepeticao.java
C:\teste>java ProgRepeticao
C:\teste>java ProgRepeticao
Input
Digite 1 repeticao for
Digite 2 para repeticao while
Digite 3 para repeticao do/while
2
OK
Cancel


Prompt de Comando - java ProgRepeticao
C:\teste>javac ProgRepeticao.java
C:\teste>java ProgRepeticao
Message
Tabuada do 2 pelo while:
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
OK


Prompt de Comando - java ProgRepeticao
C:\teste>javac ProgRepeticao.java
C:\teste>java ProgRepeticao
Input
Digite 1 repeticao for
Digite 2 para repeticao while
Digite 3 para repeticao do/while
3
OK
Cancel


Prompt de Comando - java ProgRepeticao
C:\teste>javac ProgRepeticao.java
C:\teste>java ProgRepeticao
Message
Tabuada do 2 pelo do/while:
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
OK

Atividade extra

Assista ao filme “O círculo” Adaptação do best-seller homônimo, escrito por Dave Eggers. A história parte da realização do grande sonho de Mãe (Emma Watson), que é trabalhar na maior empresa de tecnologia do mundo, O Círculo. Fundada por Eamon Bailey (Tom Hanks), a organização tem como principal produto o SeeChange, uma pequena câmera que permite aos usuários compartilharem detalhes de suas vidas com o mundo. Conforme vai subindo na hierarquia d’O Círculo, Mãe é incentivada por Bailey a viver sua vida com total transparência. Porém, quando todos estão assistindo, ninguém está realmente seguro.

Referência Bibliográfica

- FERRARI, F.; CECHINAL, C. **Introdução a Algoritmos e Programação.** Disponível em: https://docplayer.com.br/76000-Introducao-a-algoritmos-e-programacao.html Último acesso em: Julho de 2021.

- FERREIRA, Aurélio Buarque de Holanda. **Dicionário Eletrônico Aurélio Século XXI**. Rio de Janeiro: Editora Nova Fronteira e Lexikon Informática, 1999.

- PUGA, S.; RISSETTI, G. **Lógica de programação e estruturas de dados, com aplicações em Java**. Pearson: 2016.

- RIBEIRO, J. A. **Introdução à programação e aos algoritmos**. 1. ed. Rio de Janeiro: LTC, 2019

Atividade Prática – Aula 15

Título da Prática: Estrutura de Controle

Aulas Envolvidas nesta Prática: Estrutura de Controle no Java

Objetivos: Praticar lógica de programação e desenvolvimento de programas.

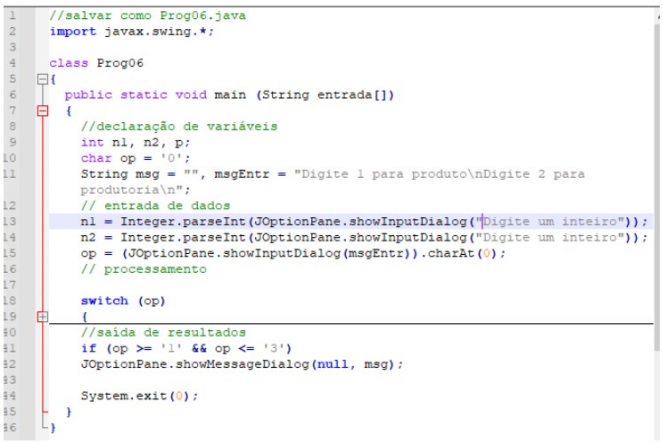
Materiais, Métodos e Ferramentas:
Para realizar este exercício, vamos utilizar Bloco de Notas e Prompt de Comando para criar e testar o programa proposto no desenvolvimento da prática em questão.

Atividade Prática
Desenvolva um programa em Java que declara variáveis inteiras, char e String, receba dois números inteiros e uma opção, calcula o produto dos dois números se eles forem positivos (ex.: p = n1 * n2), calcula a produtória do primeiro número, o número de vezes do segundo e mostra

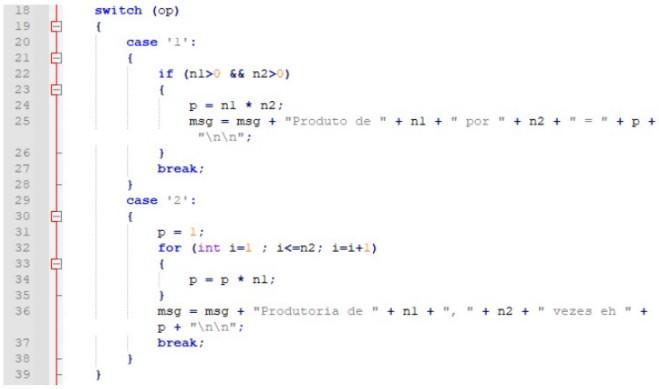
as informações (ex.: p = p * n1). Usar estruturas de decisão e de múltipla escola.

Após desenvolver seu código conforme a descrição acima, copie e cole na caixa de texto (a resposta da Atividade Prática sempre será em código (Java)).

Gabarito Atividade Prática



```
//salvar como Prog06.java
import javax.swing.*;

class Prog06
{
  public static void main (String entrada[])
  {
    //declaração de variáveis
    int n1, n2, p;
    char op = '0';
    String msg = "", msgEntr = "Digite 1 para produto\nDigite 2 para
    produtoria\n";
    // entrada de dados
    n1 = Integer.parseInt(JOptionPane.showInputDialog("Digite um inteiro"));
    n2 = Integer.parseInt(JOptionPane.showInputDialog("Digite um inteiro"));
    op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);
    // processamento

    switch (op)
    {
    //saída de resultados
    if (op >= '1' && op <= '3')
    JOptionPane.showMessageDialog(null, msg);

    System.exit(0);
  }
}
```



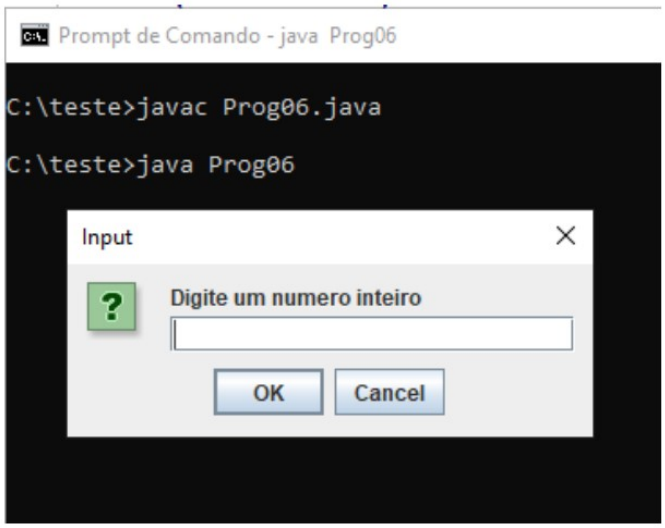
```
switch (op)
{
    case '1':
    {
        if (n1>0 && n2>0)
        {
            p = n1 * n2;
            msg = msg + "Produto de " + n1 + " por " + n2 + " = " + p +
             "\n\n";
        }
        break;
    }
    case '2':
    {
        p = 1;
        for (int i=1 ; i<=n2; i=i+1)
        {
            p = p * n1;
        }
        msg = msg + "Produtoria de " + n1 + ", " + n2 + " vezes eh " +
        p + "\n\n";
        break;
    }
}
```

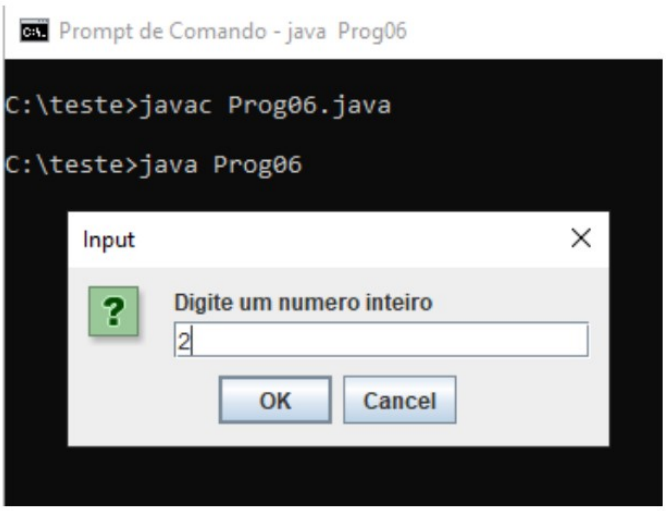
//salvar como Prog06.java

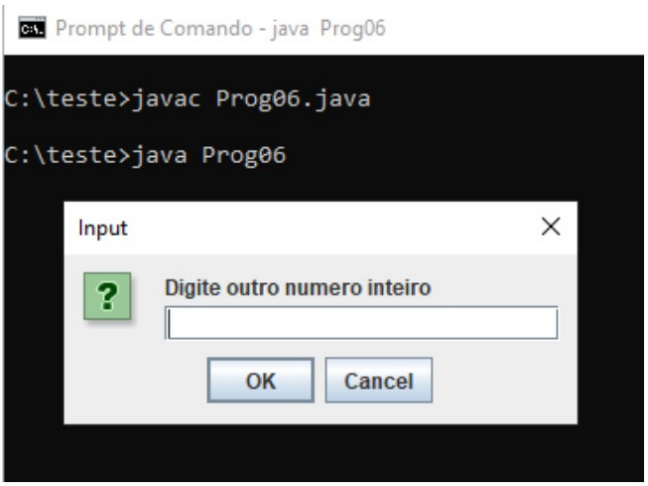
import javax.swing.*;

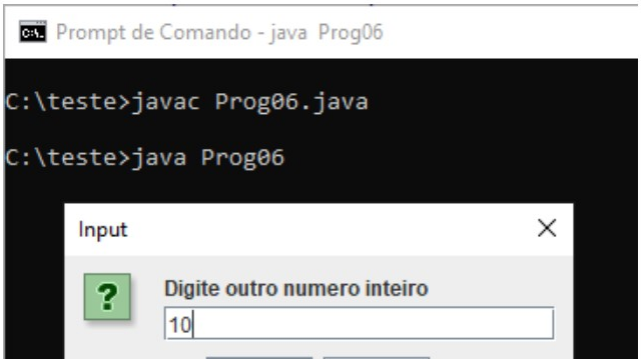
```
class Prog06
{
  public static void main (String entrada[])
  {
    //declaração de variáveis
    int n1, n2, p;
    char op = '0';
       String msg = "", msgEntr = "Digite 1 para produto\nDigite 2 para
produtoria\n";
    // entrada de dados
          n1 = Integer.parseInt(JOptionPane.showInputDialog("Digite um
inteiro"));
          n2 = Integer.parseInt(JOptionPane.showInputDialog("Digite um
inteiro"));
    op = (JOptionPane.showInputDialog(msgEntr)).charAt(0);
    // processamento


    switch (op)
    {
       case '1':
       {
          if (n1>0 && n2>0)
          {
             p = n1 * n2;
             msg = msg + "Produto de " + n1 + " por " + n2 + " = " + p + "\n\n";
          }
          break;
```

```
        }
        case '2':
        {
            p = 1;
            for (int i=1 ; i<=n2; i=i+1)
            {
                p = p * n1;
            }
             msg = msg + "Produtoria de " + n1 + ", " + n2 + " vezes eh " + p +
"\n\n";
            break;
        }
    }
    //saída de resultados
    if (op >= '1' && op <
```




Prompt de Comando - java Prog06
C:\teste>javac Prog06.java
C:\teste>java Prog06
Input
Digite um numero inteiro
2
OK
Cancel


Prompt de Comando - java Prog06
C:\teste>javac Prog06.java
C:\teste>java Prog06
Input
Digite outro numero inteiro
OK
Cancel


Prompt de Comando - java Prog06
C:\teste>javac Prog06.java
C:\teste>java Prog06
Input
Digite outro numero inteiro
10